101 Things You Can Do with Non-Classical Planning

Sheila A. McIlraith
(Joint work with Jorge A. Baier \textsuperscript{2} and Shirin Sohrabi\textsuperscript{3})
Department of Computer Science
University of Toronto

\textsuperscript{2} - Pontificia Universidad Católica de Chile, Santiago, Chile
\textsuperscript{3} - IBM T.J. Watson Research Lab, NY, USA

Originally presented at
Symposium in Honor of Dana Nau

Represented in CSC2542
February 7, 2019
...some great graduate students...

Jorge Baier       Shirin Sohrabi       Christian Fritz       Tran Cao Son

Meghyn Bienvenu   Aws Al barghouthis   Niloofar Razavi

Nathan Robinson
Yixiao Wang
Sotirios Liaskos
Given:

- Initial state (fully described)
- Goal condition (condition about final state)
- Domain description (deterministic)

Task:

- Find any plan
Non-classical planning can take many forms:

Given:

- Initial state (partially described)
- Goal condition (temporally extended goal and/or preferences)
- Domain description (non-deterministic)

Task:
- Find a (high quality) plan
Many tasks that involve reasoning about dynamical systems can be cast as non-classical planning problems (and indeed many compelling real-world planning applications are non-classical).
Many tasks that involve reasoning about dynamical systems can be cast as non-classical planning problems (and indeed many compelling real-world planning applications are non-classical).

Many non-classical planning problems can be effectively solved using state-of-the-art classical planning techniques.
Many tasks that involve reasoning about dynamical systems can be cast as non-classical planning problems. E.g.,

- (customized) web service composition
- synthesis of business processes
- verification & counter-example generation
- diagnosis of dynamical systems
- explanation generation
- stakeholder goal modeling
- test generation for concurrent programs
- network topology transformation
- ...

and at least 94 other tasks!
## Example 1 - Diagnosis

### Observations

I started my car this morning; drove to work; on the way to work I bought $5 worth of gas; I hit a pothole; the radio said it was -20 Celsius; I parked outside. At noon, I picked up my bag from the trunk of the car. At the end of the day, my car would not start. I checked the radio and it was still working.

<table>
<thead>
<tr>
<th>Event</th>
<th>Start-car</th>
<th>Drive-to-work</th>
<th>Hit-pothole</th>
<th>Bought-gas</th>
<th>Arrive-work</th>
<th>Open-trunk</th>
<th>Turn-on-radio</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>on(radio)</td>
<td>temp(-20)</td>
<td>not start(car)</td>
<td>on(radio)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>start-car</th>
<th>drive-to-work</th>
<th>hit-pothole</th>
<th>bought-gas</th>
<th>arrive-work</th>
<th>open-trunk</th>
<th>turn-on-radio</th>
</tr>
</thead>
<tbody>
<tr>
<td>on(radio)</td>
<td>temp(-20)</td>
<td>not start(car)</td>
<td>on(radio)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example 1 - Diagnosis

Observations

I started my car this morning; drove to work; on the way to work I bought $5 worth of gas; I hit a pothole; the radio said it was -20 Celsius; I parked outside. At noon, I picked up my bag from the trunk of the car. At the end of the day, my car would not start. I checked the radio and it was still working.

What’s the explanation for my car not starting?

- Battery died
- Punctured gas tank, then ran out of gas
- Starter motor broke
Dynamical Diagnosis

- Initial State
- Goal State
- Transition System

\{ \text{Observations} \ (i.e., \text{multiple partial states}) \}

\text{System Description}

\( obs_1 \quad obs_2 \quad obs_3 \quad \ldots \quad obs_n \)
Diagnosis as Non-Classical Planning

Dynamical Diagnosis

- Initial State
- Goal State
- Transition System

\[
\text{Observations} \quad \{\text{i.e., multiple partial states}\}
\]

System Description

\[
\text{Diagnosis} = \{\text{Assumptions}, \{a_1, a_2, a_3, \ldots, a_k\}\}
\]
Example 2 - Synthesis of Business Processes

- PrePay
- Invoice
- Out Source
- Manufacture
- Process Payment
- Fill Order
- Process Order
- Ship Goods
Key Enabler

It is the **richness of the statement of the planning objective** that enables us to view 101 (and more) diverse endeavours as instances of non-classical planning problems.
It is the **richness of the statement of the planning objective** that enables us to view 101 (and more) diverse endeavours as instances of non-classical planning problems.

These objectives can be specified via
- Hierarchical Task Networks (HTN)
Key Enabler

It is the **richness of the statement of the planning objective** that enables us to view 101 (and more) diverse endeavours as instances of non-classical planning problems.

These objectives can be specified via

- Hierarchical Task Networks (HTN)
- Golog
Key Enabler

It is the **richness of the statement of the planning objective** that enables us to view 101 (and more) diverse endeavours as instances of non-classical planning problems.

These objectives can be specified via

- Hierarchical Task Networks (HTN)
- Golog
- Linear Temporal Logic (LTL)
Key Enabler

It is the **richness of the statement of the planning objective** that enables us to view 101 (and more) diverse endeavours as instances of non-classical planning problems.

These objectives can be specified via

- Hierarchical Task Networks (HTN)
- Golog
- Linear Temporal Logic (LTL)

and in the case of LTL, as either **temporally extended goals** or **preferences**.
Historically, planning with HTN, Golog, and LTL has required special-purpose planners.
Many non-classical planning problems can be effectively solved using state-of-the-art classical planning techniques. (Baier’s Thesis)
Why Planning Technology?

Is this a hammer looking for a nail?
Why Planning Technology?

Is this a hammer looking for a nail?

- Obvious isomorphism
  - large search spaces
  - large transition function
Is this a hammer looking for a nail?

- **Obvious isomorphism**
  - large search spaces
  - large transition function

- **Great advances in classical planning in last 15 years**
  - compact state representations
  - implicit state transition function representation
  - highly optimized search techniques
    - SAT-based planning (optimal makespan)
    - heuristic search

- **Continuing advances in cost-optimal planning**
Approach: Reformulate a non-classical planning problem in order to apply directly or almost directly successful classical planning approaches, yielding superior performance.
Approach: Reformulate a non-classical planning problem in order to apply directly or almost directly successful classical planning approaches, yielding superior performance.

Of note: It is the objective/goal that is the main focus of the reformulation, not the domain.
The Reformulation Approach

Non-classical planning task

Reformulation Algorithm

Reformulated Planning Task

Existing Planner

Modified Planner
Reformulation Algorithms

Algorithms for:
- Temporally Extended Goals (TEGs) in LTL
- Temporally Extended Preferences (TEPs) in LTL and past LTL
- Golog-like Procedural Control
- HTN Procedural Control

Properties of Output:
- in most cases usable with a wide range of planners
- composable
Reformulation Algorithms

Algorithms for:

- Temporally Extended Goals (TEGs) in LTL
- Temporally Extended Preferences (TEPs) in LTL and past LTL
- Golog-like Procedural Control
- HTN Procedural Control

Properties of Output:

- in most cases usable with a **wide range** of planners
- **composable**
Example LTL $(\forall x). (\text{ECapital}(x) \supset \Diamond \text{at}(x)) \land \Diamond □ \text{at}(\text{Paris})$. 
Example LTL $(\forall x). (ECapital(x) \supset \diamond at(x)) \land \diamond \Box at(Paris)$. 

Resulting PFNA

$$(\neg ECapital(x) \lor at(x)) \land at(Paris)$$

Plan: $FlyTo(C_1)$;

Objects: $C_1, C_2, Paris, NYC$. $ECapital = \{C_1, C_2, Paris\}$

Initial State: $\{at(NYC)\}$
Example LTL $(\forall x). (\text{ECapital}(x) \supset \lozenge \text{at}(x)) \land \lozenge \Box \text{at}(\text{Paris})$.

Resulting PFNA

Plan: $\text{FlyTo}(C_1); \text{FlyTo}(C_2);$

Objects: $C_1, C_2, \text{Paris}, \text{NYC}$. $\text{ECapital} = \{C_1, C_2, \text{Paris}\}$

Initial State: $\{\text{at}(\text{NYC})\}$
Example LTL \((\forall x). (\text{ECapital}(x) \supset \Diamond \text{at}(x)) \land \Diamond \Box \text{at}(Paris))\).

Resulting PFNA

\[ (\neg\text{ECapital}(x) \lor \text{at}(x)) \land \text{at}(Paris) \]

Plan: \textit{FlyTo}(C_1); \textit{FlyTo}(C_2); \textit{FlyTo}(Paris);

 Objects: \(C_1, C_2, Paris, NYC\). \(\text{ECapital} = \{C_1, C_2, Paris\}\)

Initial State: \(\{\text{at}(NYC)\}\)
Key Idea: Represent the PNFA within the planning domain.

We update original problem with:

- New (Classical) Goal based on the accepting condition of the automaton
- Augmented Initial State
Experimental Evaluation

**Our Reformulation** + **FF**\(\chi\)[Thiebaux, Hoffmann & Nebel, 2005]  
*versus*  
**TLPlan**[Bacchus & Kabanza, 1998]  
... on standard benchmarks extended with TEGs.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Speedup</th>
<th>Translation time</th>
<th>Solution lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZenoTravel</td>
<td>0</td>
<td>11%</td>
<td>22%</td>
</tr>
<tr>
<td>Logistics</td>
<td>6%</td>
<td>24%</td>
<td>24%</td>
</tr>
<tr>
<td>Robot</td>
<td>0</td>
<td>44%</td>
<td>33%</td>
</tr>
</tbody>
</table>

- Translation time never exceeds 15% of planning time.
- Ratio \(|\text{Automaton}_\varphi|/|\varphi|\) never exceeds 1.
- Solution lengths are comparable.
Planning for TEGs in LTL

Contributions [Baier & M, AAAI06]

1. First-order, finite logic to represent TEGs: \( f\text{-FOLTL} \)

2. Method to compactly convert TEGs into final-state goals
   - Algorithm: \( f\text{-FOLTL} \) formula \( \rightarrow \) parameterized automaton
   - Theorem: Algorithm is correct
   - Theorem: Translation worst-case exponential
   - Property: Simplification and parameterization avoids worst-case blowup in practice.
   - Representation of the automaton in a planning problem

3. Experimental evaluation: showing orders-of-magnitude improvements over traditional approaches.
We also have results for Golog, HTN and LTL TEP reformulation.
**Contributions** [Baier, Bacchus & M, AIJ09]

1. Compilation of TEPs in PDDL3 to final-state preferences.

2. Branch-and-bound algorithm that finds plans of increasing quality.


4. Formal analysis:
   - conditions for optimality and $k$-optimality.

5. Implemented system: HPlan-P (**IPC award winner!**)

6. Experimental analysis that shows that:
   - Heuristic and pruning are key to good performance
   - Our planner is more consistent than the competition winner
Contributions [Baier, Fritz & M, ICAPS07, KR08]

1. Golog-based procedural control language for Planning
   - PDDL-based semantics for the language

2. Compilation of Golog-like control into classical planning
   - Theorem: Translation is correct
   - Theorem: Translation is polynomial

3. Experimental analysis showing that underconstrained control provides superior performance
...now back to our 1st Claim and to Diagnosis & Explanation Generation
Characterizing Explanations

Definition (Explanation)
Given a system \( \Sigma = (F, A, I) \), and an observation \( \varphi \), expressed in LTL an explanation is a tuple \((H, \alpha)\), where

1. \( H \) is a set of clauses over \( F \) st. \( I \cup H \) is satisfiable, \( I \not\models H \),
2. \( \alpha = a_0a_1 \ldots a_n \), a sequence of actions in \( A \) st. \( \alpha \) satisfies \( \varphi \) in the system \( \Sigma_A = (F, A, I \cup H) \).

Definition (Optimal Explanation)
Given a system \( \Sigma \), \( E \) is an optimal explanation for observation \( \varphi \) iff

1. \( E \) is an explanation for \( \varphi \), and
2. there does not exist another explanation \( E' \) for \( \varphi \) st \( E' \prec E \).
Proposition

Given a dynamical system $\Sigma$ and an observation formula $\varphi$, then

$$(H, \alpha = a_0a_1\ldots a_n)$$

is an explanation

iff

$\alpha$ is a plan for conformant planning problem

$P = ((F, A, I \cup H), \varphi)$

where $I \cup H$ is satisfiable and $\varphi$ is a temporally extended goal.
Relationship to Planning

**Proposition**

Given a dynamical system $\Sigma$ and an observation formula $\varphi$, then $(H, \alpha = a_0a_1 \ldots a_n)$ is an explanation iff

$\alpha$ is a plan for **conformant** planning problem $P = ((F, A, I \cup H), \varphi)$

where $I \cup H$ is satisfiable and $\varphi$ is a temporally extended goal.

**Theorem**

Given a dynamical system $\Sigma$ and a temporally extended formula $\varphi$, explanation existence is PSPACE-complete.
Relationship to Planning

Proposition

Given a dynamical system $\Sigma$ and an observation formula $\varphi$, then

$$(H, \alpha = a_0a_1 \ldots a_n) \text{ is an explanation}$$

iff

$\alpha$ is a plan for **conformant** planning problem

$P = ((F, A, I \cup H), \varphi)$

where $I \cup H$ is satisfiable and $\varphi$ is a temporally extended goal.

Theorem

Given a dynamical system $\Sigma$ and a temporally extended formula $\varphi$, explanation existence is PSPACE-complete.

Theorem

It is possible to find explanations using classical planning.

Generation of an optimal explanation corresponds to preference-based planning problem.
Challenges

1. How to encode observations
   - as temporally extended goal
   - by compiling them away

2. How to generate preferred diagnoses
   - using a preference-based planner
   - using an optimal cost planner

3. How to plan with incomplete initial state
Experimental Setup

- Used benchmark computer problem (20 components, 5x4 grid) defined by [Grastien et al., 2007].

- 2 sets of 20 problems:
  - First set: totally ordered observations.
  - Second set: partially ordered observations.
  - Both: problem $i$ has a minimal diagnosis with $i$ faulty actions.

- Exploited state-of-the-art SAT-based and heuristic search planners.

- 600 second time out.
Comparison to State of the Art

Can planners do as well as the state of the art in diagnosis?

Results reported from [Grastien et al, 2007]
## Comparison to State of the Art

Can planners do as well as the state of the art in diagnosis?

<table>
<thead>
<tr>
<th>#</th>
<th>MiniSat</th>
<th>FF</th>
<th>Lama</th>
<th>Satplan</th>
<th>MiniSat</th>
<th>FF</th>
<th>Lama</th>
<th>Satplan</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.03</td>
<td>0.00</td>
<td>0.00</td>
<td>0.30</td>
<td>0.04</td>
<td>0.00</td>
<td>0.01</td>
<td>0.10</td>
</tr>
<tr>
<td>3</td>
<td>0.26</td>
<td>0.02</td>
<td>0.05</td>
<td>2.14</td>
<td>0.14</td>
<td>0.03</td>
<td>0.07</td>
<td>0.89</td>
</tr>
<tr>
<td>5</td>
<td>1.90</td>
<td>0.03</td>
<td>0.14</td>
<td>6.38</td>
<td>9.70</td>
<td>0.08</td>
<td>0.36</td>
<td>2.27</td>
</tr>
<tr>
<td>7</td>
<td>9.10</td>
<td>0.16</td>
<td>0.40</td>
<td>12.56</td>
<td>350.00</td>
<td>24.98</td>
<td>12.93</td>
<td>4.10</td>
</tr>
<tr>
<td>9</td>
<td>61.00</td>
<td>0.10</td>
<td>0.85</td>
<td>OM</td>
<td>63.00</td>
<td>37.73</td>
<td>OT</td>
<td>11.68</td>
</tr>
<tr>
<td>11</td>
<td>3.90</td>
<td>0.25</td>
<td>1.22</td>
<td>OM</td>
<td>OT</td>
<td>586.76</td>
<td>OT</td>
<td>25.23</td>
</tr>
<tr>
<td>13</td>
<td>8.70</td>
<td>0.20</td>
<td>1.67</td>
<td>OM</td>
<td>OT</td>
<td>32.22</td>
<td>OT</td>
<td>35.96</td>
</tr>
<tr>
<td>15</td>
<td>18.70</td>
<td>0.17</td>
<td>4.99</td>
<td>OM</td>
<td>OT</td>
<td>16.76</td>
<td>OM</td>
<td>36.71</td>
</tr>
<tr>
<td>17</td>
<td>30.00</td>
<td>0.22</td>
<td>4.95</td>
<td>OM</td>
<td>OT</td>
<td>128.36</td>
<td>OM</td>
<td>96.15</td>
</tr>
<tr>
<td>19</td>
<td>66.00</td>
<td>0.22</td>
<td>8.10</td>
<td>OM</td>
<td>OT</td>
<td>169.34</td>
<td>OM</td>
<td>99.12</td>
</tr>
</tbody>
</table>

Results reported from [Grastien et al, 2007]
How effective are cost optimizing planners in computing a minimal fault diagnosis?

<table>
<thead>
<tr>
<th>#</th>
<th>LAMA</th>
<th>SGPlan\textsubscript{6}</th>
<th>Baseline</th>
<th>HPLAN-P</th>
<th>LAMA</th>
<th>Baseline</th>
<th>HPLAN-P</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.01</td>
<td>0.08</td>
<td>0.01</td>
<td>0.29</td>
<td>0.02</td>
<td>0.04</td>
<td>0.8</td>
</tr>
<tr>
<td>3</td>
<td>0.07</td>
<td>0.14</td>
<td>1.93</td>
<td>25.25</td>
<td>0.11</td>
<td>47.09</td>
<td>2.14</td>
</tr>
<tr>
<td>5</td>
<td>0.21</td>
<td>0.25</td>
<td>24.39</td>
<td>107.21</td>
<td>0.97</td>
<td>OT</td>
<td>579.85</td>
</tr>
<tr>
<td>7</td>
<td>0.64</td>
<td>7.64</td>
<td>54.53</td>
<td>449.53</td>
<td>17.95</td>
<td>OT</td>
<td>30.37</td>
</tr>
<tr>
<td>9</td>
<td>1.62</td>
<td>6.1</td>
<td>73.1</td>
<td>OT</td>
<td>334.01</td>
<td>OM</td>
<td>286.16</td>
</tr>
<tr>
<td>13</td>
<td>3.82</td>
<td>7.97</td>
<td>80.74</td>
<td>OT</td>
<td>OM</td>
<td>OM</td>
<td>OT</td>
</tr>
<tr>
<td>15</td>
<td>16.28</td>
<td>12.49</td>
<td>66.89</td>
<td>OT</td>
<td>OM</td>
<td>OM</td>
<td>OT</td>
</tr>
<tr>
<td>17</td>
<td>20.17</td>
<td>10.54</td>
<td>101.15</td>
<td>OT</td>
<td>OM</td>
<td>OM</td>
<td>OT</td>
</tr>
<tr>
<td>19</td>
<td>29.54</td>
<td>10.87</td>
<td>139.98</td>
<td>OT</td>
<td>OM</td>
<td>OM</td>
<td>OT</td>
</tr>
</tbody>
</table>
Many tasks that involve reasoning about dynamical systems can be cast as non-classical planning problems and solved effectively using state-of-the-art planning techniques.
And here are 94 of the 101 Things You Can Do with Non-Classical Planning:

- (customized) web service composition [M, Son. KR02], [Sohrabi, M. various]
- synthesis of business processes [Sohrabi, M. various]
- verification & counter-example generation [Albarghouthi, Baier, M. VVPS@ICAPS09]
- diagnosis of dynamical systems [Sohrabi, Baier, M. KR10]
- explanation generation [Sohrabi, Baier, M. AAAI11]
- stakeholder goal modeling [Liaskos, M, Sohrabi, Mylopoulos. REJ11]
- test generation for concurrent programs [Razavi, Farzan, M. VVPS@ICAPS11]
- network topology transformation [Young, Nathanson, et al. CP4PS@AAA12]

... 

In most of the above cases, planning technology significantly outperformed the state of the art!